REMARKS

By this amendment, Claims 1, 9, 10, 18, 39, 47, 48, and 56 are amended, Claims 8, 17, 46, and 55 are canceled, and no claims are added. Hence, Claims 1-4, 6, 7, 9-13, 15, 16, 18-20, 39-42, 44-51, and 53-58 are pending in the application. The amendments to the claims as indicated herein do not add any new matter to this application.

Each issue raised in the Office Action mailed January 2, 2008 is addressed hereinafter.

I. SUMMARY OF THE INTERVIEW

The Examiner is thanked for the in-person interview conducted with a representative of the Applicants on April 8, 2008. The parties discussed Claim 1, operation of embodiments of the disclosure, the Idicula reference, and operating system issues. The Examiner indicated that clarification of how the access identifier is used could move the case forward. No agreement was reached. An amendment that recites how the access identifier is used is included herein.

II. SUMMARY OF THE REJECTIONS

Claims 1-4, 6-13, and 15-17 stand rejected under 35 U.S.C. § 101 as allegedly directed to non-statutory subject matter. This rejection is respectfully traversed.

Claims 1-4, 7-11, 13, 16-20, 39-42, 45-49, 51, and 54-58 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over U.S. Patent No. 6,950,822 issued to Idicula et al. ("*Idicula*") in view of U.S. Patent No. 5,911,143 issued to Deinhart et al. ("*Deinhart*"). This rejection is respectfully traversed.

Claims 6, 15, 44, and 53 stand rejected under 35 U.S.C. 103(a) as allegedly being unpatentable over *Idicula* in view of *Deinhart*, and further in view of U.S. Patent No. 6,233,576 issued to Lewis ("*Lewis*"). This rejection is respectfully traversed.

Claims 12 and 50 stand rejected under 35 U.S.C. 103(a) as allegedly being unpatentable over *Idicula* in view of *Deinhart*, and further in view of Official Notice. This rejection is respectfully traversed.

III. ISSUES NOT RELATING TO THE CITED ART

Claims 1-4, 6-13, and 15-17 stand rejected under 35 U.S.C. § 101 as allegedly directed to non-statutory subject matter. The Examiner failed to address Applicants' remarks in the response filed on October 15, 2007.

In the Final Office Action and the present Office Action, the Examiner asserts, without support, that "both the apparatus and method claims may be considered to be software, per se, since both claims fail to be integrated into a computer hardware system for execution" (page 2). Neither the United States Code (particularly 35 U.S.C. § 101), nor the Code of Federal Regulations, nor the Manual of Patent Examining Procedure requires a method claim or an apparatus means plus function claim to recite a computer hardware system. A process is one of the explicitly-enumerated statutory classes under 35 U.S.C. § 101. A means plus function claim is explicitly provided for under 35 U.S.C. § 112(6). If it is the Office Action's position that the steps recited in Claims 1 and 10 may be mental steps, then it is also respectfully noted that the claims recite steps that cannot be mentally performed. For example, Claim 1 explicitly recites "creating and storing in a filesystem of an Operating System a plurality of files that each represents a different resource of the plurality of resources" (emphasis added). Furthermore, Claims 1 and 10 presently recite a "computer-implemented method." Therefore, the method is implemented by a computer.

In the Final Office Action and the present Office Action, the Examiner also incorrectly asserts that Claim 6 recites "only if the permission bit allows the operation." Instead, Claim 6

recites "only **when** the permission bit allows the operation" (emphasis added). This amendment was made as suggested by the Examiner in the Final Office Action mailed July 16, 2007.

The Examiner also incorrect asserts that because a resource operation may be not performed (due to optional language in the claim), the method Claims 6-7 and 15-16 do not produce a "useful, concrete, and tangible" result. It is respectfully noted that these claims produce a useful, concrete, and tangible result because an operation should not be performed when the permission bit of a file indicates that the operation should not be performed, even when a user desires the performance of the operation.

The Examiner also incorrectly asserts that Claims 44, 45, and other claims include "if" statements (page 3). Even though "if" statements in claims are not improper, none of the pending claims includes the word "if."

IV. ISSUES RELATING TO THE CITED ART

Claims 1-4, 7-11, 13, 16-20, 39-42, 45-49, 51, and 54-58 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over *Idicula* in view of *Deinhart*.

A. CLAIM 1

Claim 1 recites:

> A computer-implemented method for controlling access to a resource of a plurality of resources, the method comprising the steps of:
> **creating and storing in a filesystem of an Operating System a plurality of files that each represents a different resource of the plurality of resources;**
> **assigning an access value to a file attribute of a file that represents the resource, wherein the file attribute is used by the Operating System to manage file access, wherein the access value corresponds to a combination of a particular role and the resource;**
> receiving user-identifying information from a user requesting access to the resource, wherein the user-identifying information comprises a role associated with the user, wherein the role is determined from a user identifier uniquely associated with the user and from a group identifier associated with a group that includes the user;

receiving a resource identifier associated with the resource;

creating an access identifier based on the user-identifying information and the resource identifier, wherein the access identifier is formatted as a file attribute that is used by the Operating System to manage file access;

calling the Operating System to perform a file operation on the file, wherein calling the Operating System includes providing the access identifier to the Operating System; and

granting the user access to the resource only when the Operating System call successfully performs the file operation, **wherein the Operating System call successfully performs the file operation if the access identifier matches the access value;**

wherein the file operation on the file representing the resource is selected from a group consisting of opening the file, closing the file, deleting the file, reading from the file, writing to the file, executing the file, appending to the file, reading a file attribute, and writing a file attribute. (emphasis added)

1. Discussion of Claim 1

In an embodiment, resources may be a software application, software application message, server, router, switch, file, directory, etc. (Specification, paragraphs 27 and 28). A file is created and stored in a filesystem of an Operating System. The file represents a particular resource of a plurality of resource. An access value is assigned to a file attribute of that file. The file attribute is used by the Operating System to manage file access. The access value corresponds to a combination of a particular role and the resource.

Subsequently, user-identifying information is received from a user who is requesting access to that resource. A resource ID associated with the resource is also received, e.g., from the user. Subsequently, an access ID is created based on the user-identifying information and the resource ID. The access ID is formatted as a file attribute that is used by the Operating System to gain access to the file. The Operating System is then called to perform a file operation (e.g., open, write, delete) on the file that represents the resource. This is done by providing the access ID to the Operating System. The user is granted access to the particular resource when the

Operating System call successfully performs the file operation. The Operating System call successfully performs the file operation if the access identifier matches the access value.

Because Operating System calls are used to attempt access to a file, a complex security mechanism is avoided and application software developers are not required to learn a specific API and to write code against it (see specification, paragraph 3). Claim 1 uses the filesystem of an Operating System (OS) to manage access to files. The OS filesystem is one of the most well-known and easy to write code systems in a computer system (see specification, paragraphs 4, 7, and 8).

2. *Discussion of* Idicula

In contrast, *Idicula* does not teach anything substantial about (1) an Operating System, (2) a filesystem, or (3) a file that represents the resource. Instead, *Idicula* teaches a technique for servicing requests for database services (Abstract). Specifically, *Idicula* teaches that rather than deleting session objects associated with a particular session when the corresponding connection ends, some of those session objects are saved and reused with new connections (col. 6, lines 6-9). Thus, the resources that would be consumed to delete and recreate session objects when connections are ended and started, respectively, are conserved (col. 6, lines 11-15).

Due to these significant differences, *Idicula* fails to teach or suggest numerous features of Claim 1. A prior art rejection is supportable only if the cited prior art or evidence clearly discloses or suggests every feature of the claims in the combination that is claimed. In the Interview, it was apparent that to at least some extent, the Office has been recharacterizing the claims in more general or abstract terms, ignoring certain expressly recited features. However, a rigorous comparison of the claims to the cited references plainly establishes allowability for at least the following reasons.

3. The cited art fails to teach or suggest the recited access value or how the recited access identifier is used

Claim 1 is amended to recite "assigning an access value to a file attribute of a file that represents the resource, wherein the file attribute is used by the Operating System to manage file access, wherein the access value corresponds to a combination of a particular role and the resource." This access value is later used in conjunction with the recited access identifier that is created after a user requests access to a particular resource. The Operating System call successfully performs a file operation on the file that represents the particular resource if the access identifier matches the access value.

Not only has the Office Action failed to equate any element of the cited art with the recited access identifier, nothing in the cited art can be equated to the recited access value that is assigned to a file attribute of a file. As a result, the cited art must also fail to teach or suggest that an access identifier (that is created in response to a request for a resource) is matched against an access value that is assigned to a file attribute of a file that represents the resource.

4. The database of Idicula *cannot be the recited resource*

Further, Claim 1 recites: "creating and storing in a filesystem of an Operating System a plurality of files that each represents a different resource of the plurality of resources." The Office Action equates the database of *Idicula* with the resource of Claim 1. This does not make sense because Claim 1 recites (a) a **plurality** of files that are stored in the recited filesystem and (b) each file represents a different resource of a **plurality** of resources.

5. The database session object of Idicula *cannot be the recited file*

The Office Action equates the database session object of *Idicula* with the file of Claim 1 (page 9). This is incorrect. According to *Idicula*, database session objects are stored in a

database server (see FIG. 1 of *Idicula*), **not** a filesystem, which is part of a database (see col. 2, lines 10-11).

Additionally, *Idicula* specifically refers to files and a filesystem and distinguishes files from database session objects. A portion of col. 2, lines 6-18 of *Idicula* states: "to support database requests for **hierarchical data, such as files and folders of a file system** stored in a repository within the database, the **database session object is extended to include information about a root container for the hierarchy**" (emphasis added). Thus, *Idicula* clearly indicates that database session objects are not files of a filesystem. A database session object may merely be extended to **include information about** a root container for a hierarchy of data.

Col. 5, lines 30-31 of *Idicula* further teach that files (and folders) are stored in a database. Therefore, according to *Idicula*, a database session object is **different** than a file. Also, based on the interpretation of the Office Action, a database session object would have to be stored in the filesystem referenced in col. 2, line 10 of *Idicula*. However, *Idicula* fails to teach or suggest that a database session object is stored in the referenced filesystem.

6. Idicula *fails to teach or suggest calling an Operating System according to Claim 1*

Claim 1 further recites "calling the Operating System to perform a file operation on the file by providing the access identifier to the Operating System to attempt access to the file." In response to Applicants' argument that *Idicula* fails to teach or suggest this feature of Claim 1, the Office Action states:

> Idicula discloses a system wherein the contents of a session object (e.g., session information) are checked to see if a session has already been created for a client. Upon finding that a session object associated with the client is indicated in the process state object (i.e., the file operation), the client is then allowed to received the requested database services from the database (page10).

Nowhere does *Idicula* teach or suggest that an ***operating system*** **is called**. *Idicula* specifically refers to an operating system at col. 1, lines 58-61. Even if the finding of a database session object could be considered a "file operation", *Idicula* fails to teach or suggest that the referenced operating system is called to perform the "file operation" on the database session object.

Claim 1 further recites "granting the user access to the resource only when the Operating System call successfully performs the file operation" (emphasis added). The Office Action, as indicated previously, equates the database of *Idicula* with the recited "resource" of Claim 1. The Office Action asserts that:

> a client seeking database services passes a request to an Operating System. Within the passing of the request, client information associated with the session and resource are called and passed on to the Operating System. With the client information, the Operating System attempts to read the session object, searching for the presence of an existing session for the client. Wherein the Operating System successfully finds an existing session for the client, the client is then granted access to the database" (page 10).

It is respectfully submitted that this is incorrect. As indicated previously, *Idicula* only mentions "operating system" once, at col. 1, line 59. No where does *Idicula* teach that: (1) a client passes a request to **an Operating System**; (2) client information associated with a session is passed to **an Operating System**; or (3) **an Operating System** attempts to read a database session object by searching for the presence of an existing session, as the Office Action alleges. Not only does *Idicula* distinguish between an Operating System and a database server, *Idicula* explicitly teaches that a database server performs the above steps alleged in the Office Action. For example, col. 7, lines 21-23 of *Idicula* states, "The **database server** determines whether a session has already been created for this client by checking the contents of process state object 130" (emphasis added). As another example, col. 4, lines 15-18 of *Idicula* states, "The techniques described hereafter allow a **database server** to more efficiently service more requests for database services" (emphasis added). As yet another example, col. 4, lines 42-44 of *Idicula*

states, "The database server 110 includes memory 120 on the database server host computer, which is allocated for use by the **database server** 110. The database server 110 maintains several data structures in memory 120" (emphasis added), which data structures include the database session objects and process state objects.

Based on the foregoing, *Idicula* fails to teach or suggest numerous features of Claim 1. Additionally, *Dienhart* is not used to show the features of Claim 1 that are not taught or suggested by *Idicula*. Therefore, Claim 1 is patentable over *Idicula* and *Deinhart*. Reconsideration and withdrawal of the rejection of Claim 1 under 35 U.S.C. § 103(a) is therefore respectfully requested.

B. CLAIMS 10, 18, 39, 48, AND 56

The Office Action stated the same reasons in rejecting Claims 10 and 18 to those in rejecting present Claim 1. Also, Claims 39, 48 and 56 recite features discussed above that make Claim 1 patentable over *Idicula* and *Deinhart*. Therefore, for at least the same reasons set forth above by the Applicant in connection with present Claim 1, it is respectfully submitted that each of Claims 10, 18, 39, 48 and 56 is patentable over *Idicula* and *Deinhart*.

C. DEPENDENT CLAIMS

The dependent claims not discussed thus far are dependent claims, each of which depends (directly or indirectly) on one of the independent claims discussed above. Each of the dependent claims is therefore allowable for the reasons given above for the claim on which it depends. In addition, each of the dependent claims introduces one or more additional limitations that independently render it patentable. However, due to the fundamental differences already identified, to expedite the positive resolution of this case, a separate discussion of those

limitations is not included at this time. The Applicant reserves the right to further point out the differences between the cited art and the novel features recited in the dependent claims.

V. CONCLUSIONS & MISCELLANEOUS

For the reasons set forth above, all of the pending claims are now in condition for allowance. The Examiner is respectfully requested to contact the undersigned by telephone relating to any issue that would advance examination of the present application.

A petition for extension of time, to the extent necessary to make this reply timely filed, is hereby made. If applicable, a law firm check for the petition for extension of time fee is enclosed herewith. If any applicable fee is missing or insufficient, throughout the pendency of this application, the Commissioner is hereby authorized to any applicable fees and to credit any overpayments to our Deposit Account No. 50-1302.

Respectfully submitted,
HICKMAN PALERMO TRUONG & BECKER LLP

Dated: April 23, 2008

/DanielDLedesma#57181/
Daniel D. Ledesma
Reg. No. 57,181

2055 Gateway Place Suite 550
San Jose, California 95110-1083
Telephone No.: (408) 414-1080 x229
Facsimile No.: (408) 414-1076